

INTERACTIVE MULTIMEDIA DESIGN

With an AI touch :)

Computer programming

2



Computer programming

3



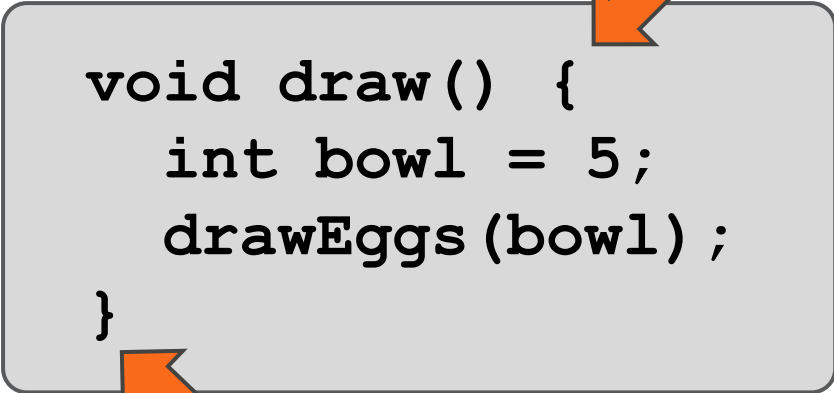
Computer programming

4



Blocks of code

5

A light gray rounded rectangle containing code. Two orange arrows point to the opening and closing curly braces of the function definition.

```
void draw() {  
    int bowl = 5;  
    drawEggs (bowl) ;  
}
```

- In computer programming we separate **blocks** of code using **brackets**
- This **block** has a name: “`void draw()`”
- This is the block of code that tells to the system what to draw on our screen when we run it

Blocks of code

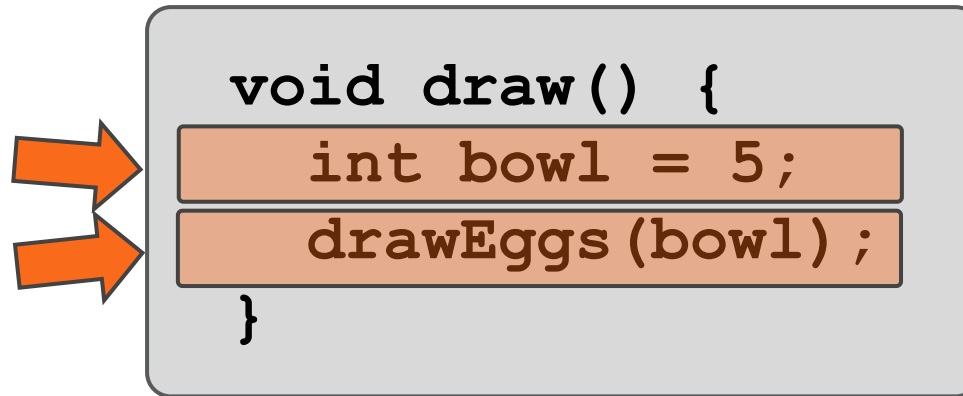
6

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- In computer programming we separate **blocks** of code using **brackets**
- This **block** has a name: “`void draw()`”
- This is the block of code that tells to the system what to draw on our screen when we run it

Statements

7



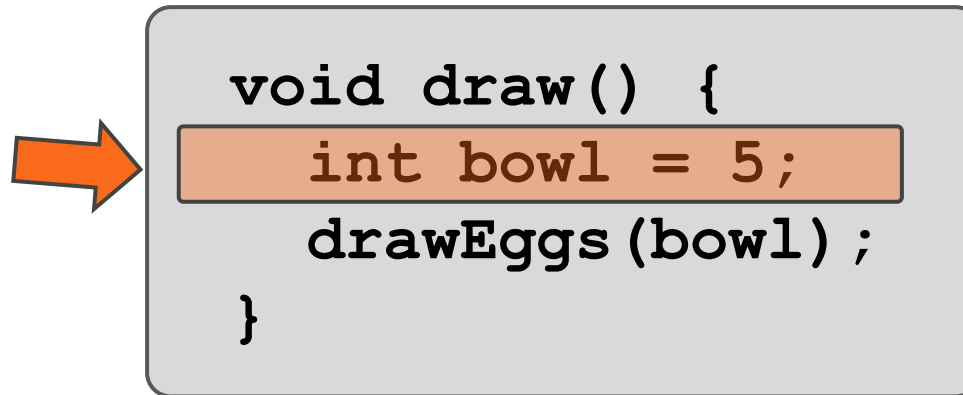
The diagram shows a code block for a function named `draw()`. The function body contains two lines of code: `int bowl = 5;` and `drawEggs(bowl);`. Each line is enclosed in a light orange rectangular box. Two orange arrows point from the left towards these two boxes, indicating that they represent individual statements within the block.

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- This **block** consists of two **statements**
- Each statement is on a separate line
- Every **statement** ends with a semicolon

Variables

8



```
void draw() {  
    int bowl = 5;  
    drawEggs (bowl) ;  
}
```

- We are going to use **variables** a lot!
- A variables is like a little **box** that keeps one piece of **information** inside
- E.g., here, bowl keeps an integer number

Variables

9

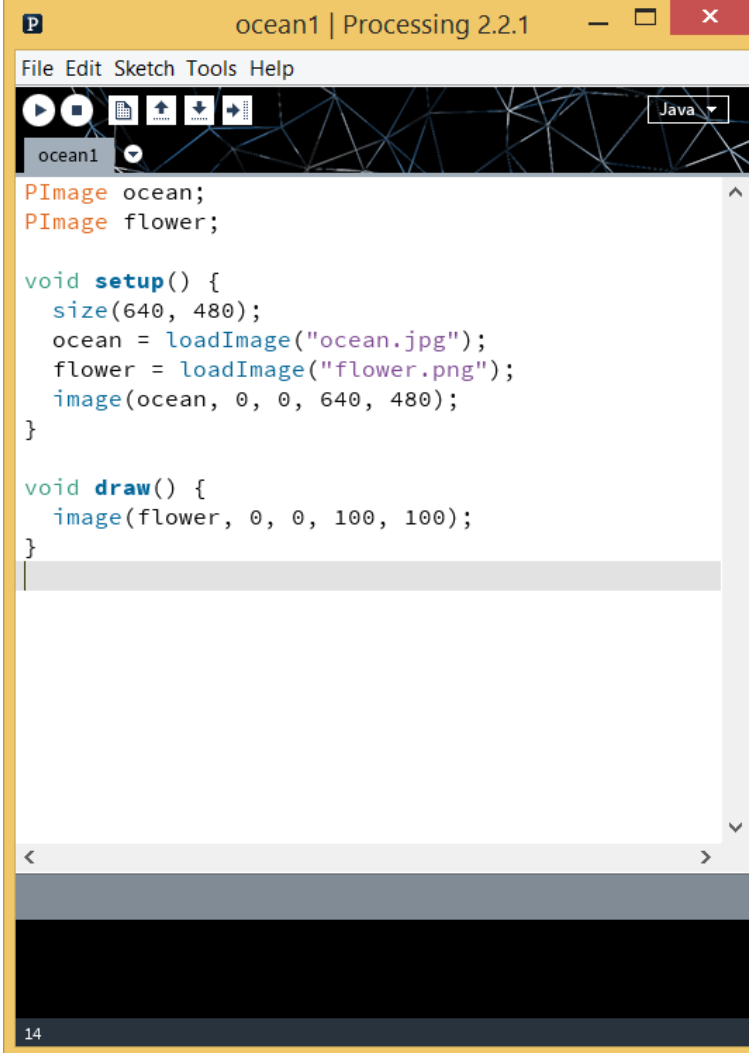
```
void draw() {  
    int bowl = 2;  
    int anotherbowl = 4;  
    bowl = anotherbowl;  
    //bowl = anotherbowl + 1;  
    //bowl = bowl + 1;  
    drawEggs (bowl) ;  
}
```

- We are going to use a lot of variables!

Magic recipes, ocean1.pde

10

- Download file ocean1.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open ocean1.pde from folder ocean1



```
ocean1 | Processing 2.2.1
File Edit Sketch Tools Help
ocean1
PImage ocean;
PImage flower;

void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

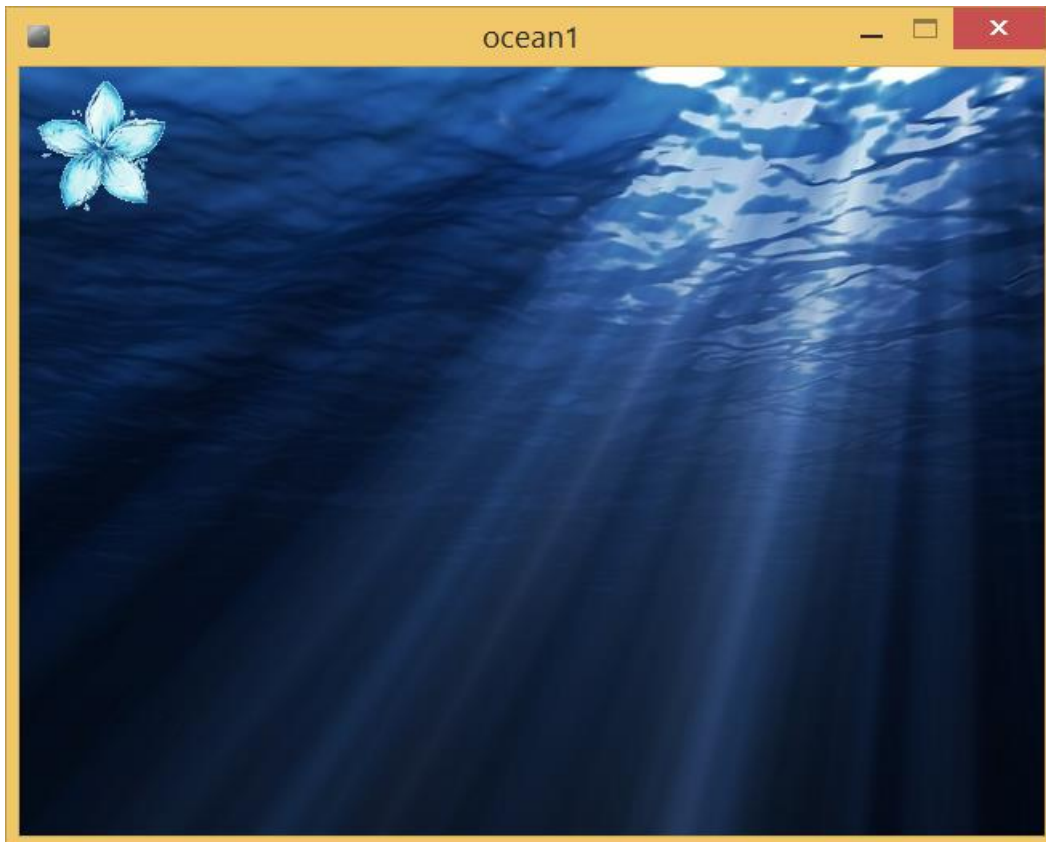
void draw() {
  image(flower, 0, 0, 100, 100);
}

14
```

Magic recipes, ocean1.pde

11

- Place two images on the window



```
ocean1 | Processing 2.2.1
File Edit Sketch Tools Help
ocean1
PImage ocean;
PImage flower;

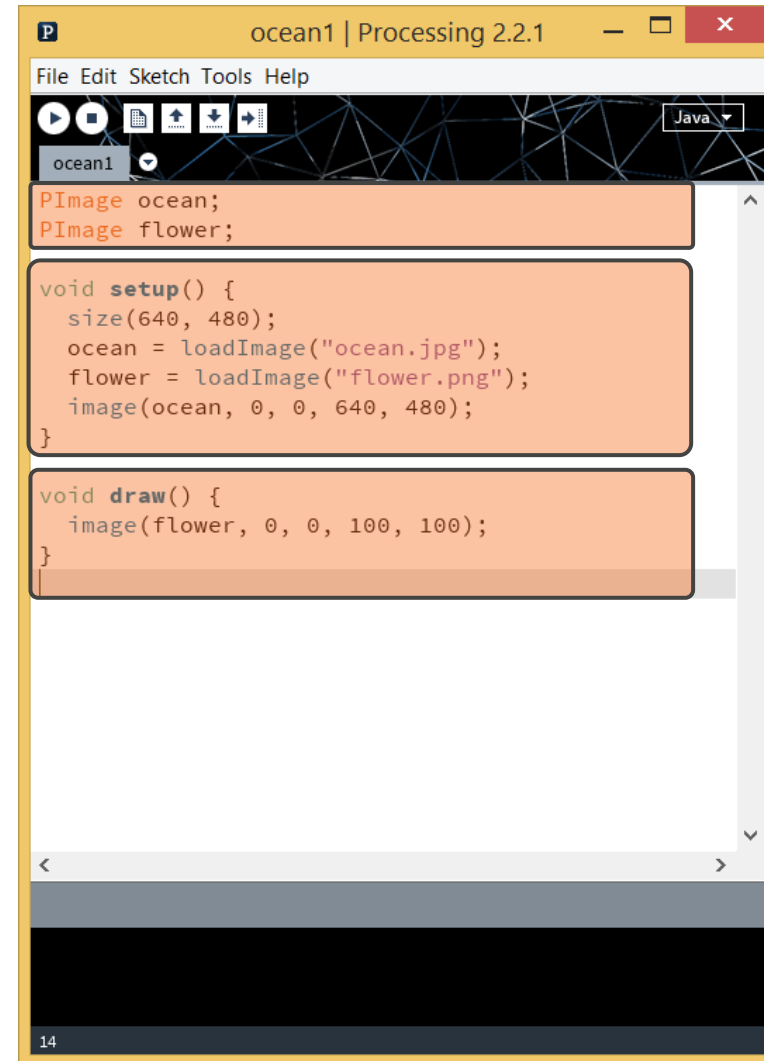
void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

void draw() {
  image(flower, 0, 0, 100, 100);
}
14
```

Magic recipes, ocean1.pde

12

- For each image file that we want to use, we need to do three things
 - ▣ Declare a variable
PImage ocean;
 - ▣ Initialize the variable to load the image file we want to use
 - ▣ Use the variable to draw the image



```
P ocean1 | Processing 2.2.1
File Edit Sketch Tools Help
ocean1
PImage ocean;
PImage flower;

void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

void draw() {
  image(flower, 0, 0, 100, 100);
}

14
```

Processing – Images

13

- `variableName = loadImage("image name");`

```
ocean = loadImage("ocean.jpg");  
flower = loadImage("flower.png");
```

- `image(img, xPosition, yPosition, width, height);`

```
image(ocean, 0, 0, 640, 480);  
image(flower, 0, 0, 100, 100);
```

Processing – Images

14

- `tint(red, green, blue, transparency)`

- Values from 0 to 255,

- E.g., green and very transparent:

```
tint(0, 255, 0, 30);
```

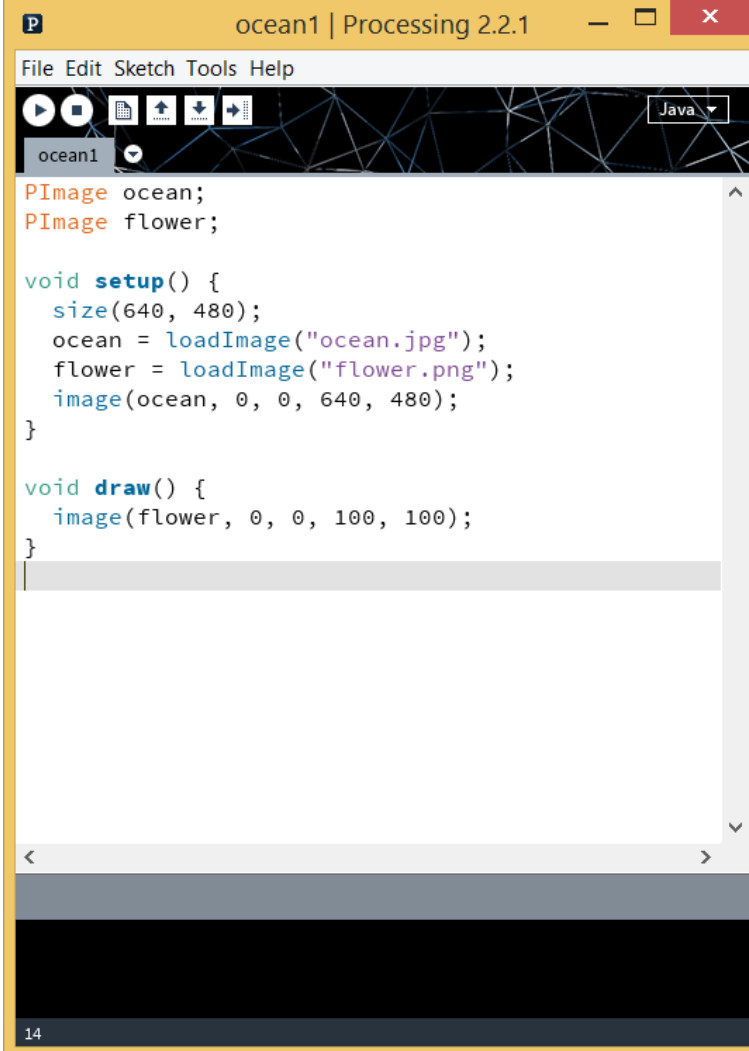
- E.g., red and no transparent:

```
tint(255, 0, 0, 255);
```

Magic recipes, ocean2.pde

15

- Download file ocean2.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open ocean2.pde from folder ocean2



```
ocean1 | Processing 2.2.1
File Edit Sketch Tools Help
ocean1
PImage ocean;
PImage flower;

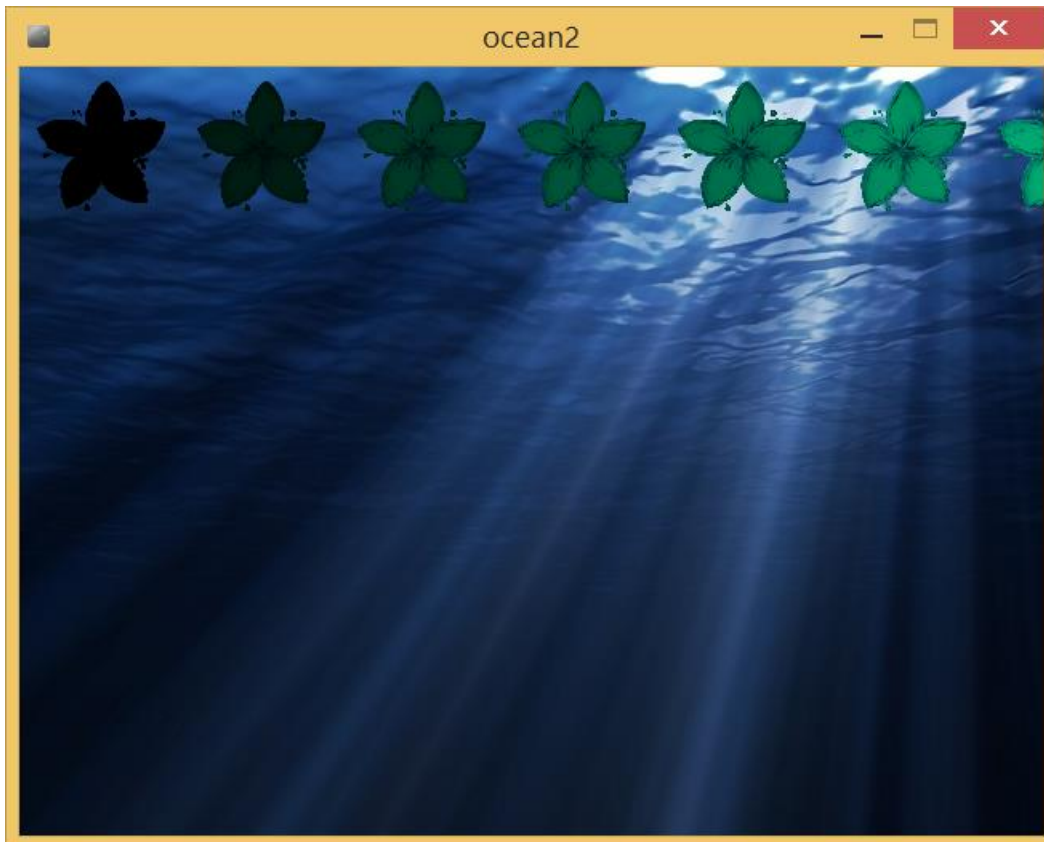
void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

void draw() {
  image(flower, 0, 0, 100, 100);
}
14
```

Magic recipes, ocean2.pde

16

- Place many different copies of one image



```
ocean2 | Processing 2.2.1
File Edit Sketch Tools Help
ocean2
PImage ocean;
PImage flower;

void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

void draw() {
  for (int i=0; i<10 ; i++){
    tint(0, 30*i, 20*i);
    image(flower, 100*i, 0, 100, 100);
  }
}
```


Processing – For loop

17

- Make processing do many things for you!

```
“For loop statement” {  
  block of code  
}
```

- Execute the block **many times!**

Processing – For loop

18

- Make processing do many things for you!

```
“For loop statement” {  
    image(flower, 100*i, 0, 100, 100);  
}
```

- Execute the block **many times!**

Processing – For loop

19

- Make processing do many things for you!

```
“For loop statement” {  
    image(flower, 100*i, 0, 100, 100);  
}
```

- **Each time replace the variable *i* with the following values, and then execute the block**
 - $i = 1$
 - $i = 2$
 - $i = 3$
 - $i = 4$

Processing – For loop

20

- Make processing do many things for you!

```
“For loop statement” {  
    image(flower, 100*i, 0, 100, 100);  
}
```

- **Each time replace the variable *i* with the following values, and then execute the block**

- `i = 1 → image(flower, 100*1, 0, 100, 100);`

- `i = 2 → image(flower, 100*2, 0, 100, 100);`

- `i = 3 → image(flower, 100*3, 0, 100, 100);`

- `i = 4 → image(flower, 100*4, 0, 100, 100);`

Processing – For loop

21

- Make processing do many things for you!

```
“For loop statement” {  
    image(flower, 100*i, 0, 100, 100);  
}
```

- **Each time replace the variable *i* with the following values, and then execute the block**

- $i = 1 \rightarrow \text{image}(\text{flower}, 100*1, 0, 100, 100);$

- $i = 2 \rightarrow \text{image}(\text{flower}, 100*2, 0, 100, 100);$

- $i = 3 \rightarrow \text{image}(\text{flower}, 100*3, 0, 100, 100);$

- $i = 4 \rightarrow \text{image}(\text{flower}, 100*4, 0, 100, 100);$

Processing – For loop

22

- Make processing do many things for you!

```
“For loop statement” {  
    image(flower, 100*i, 0, 100, 100);  
}
```

- **Each time replace the variable i with the following values, and then execute the block**
 - **Start with $i = 1$, and execute the block**
 - **Continue as long as $i < 5$**
 - **Add 1 to i , and execute the block**
 - ...

Processing – For loop

23

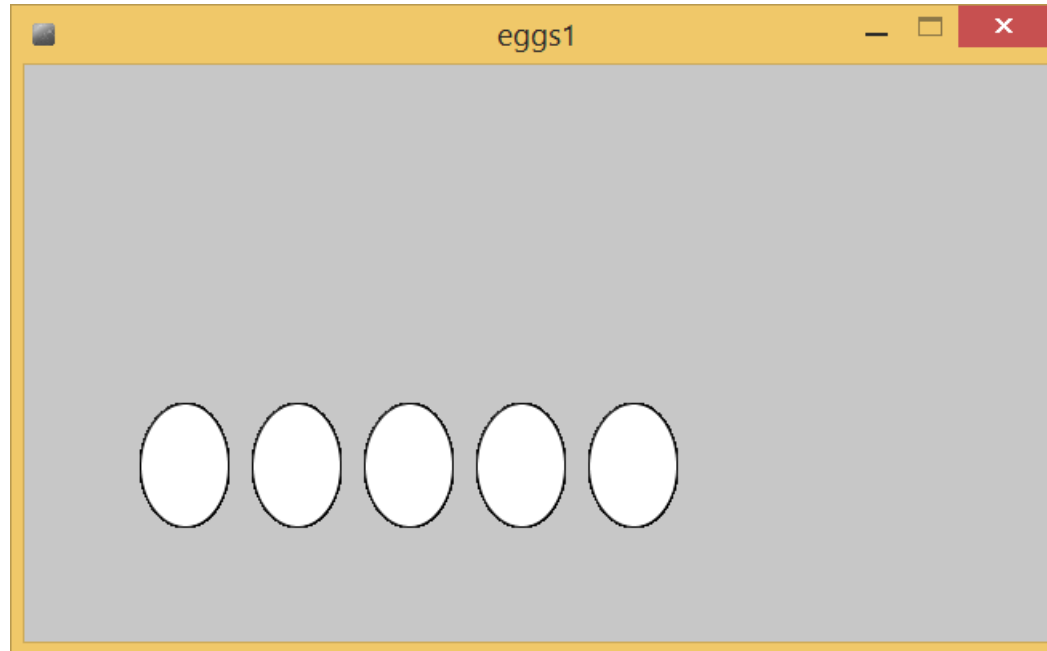
- Make processing do many things for you!

```
for (i=1; i<5; i++) {  
    image (flower, 100*i, 0, 100, 100);  
}
```

- **Each time replace the variable *i* with the following values, and then execute the block**
 - **Start with *i* = 1,** and execute the block
 - **Continue as long as *i* < 5**
 - **Add 1 to *i*,** and execute the block
 - ...

Magic recipes, eggs1.pde

24



- Remember the function that drew eggs on the screen? It was just a simple for loop, like the one we did with flowers here, but with `ellipse()` instead ;)

Magic recipes, eggs1.pde

25

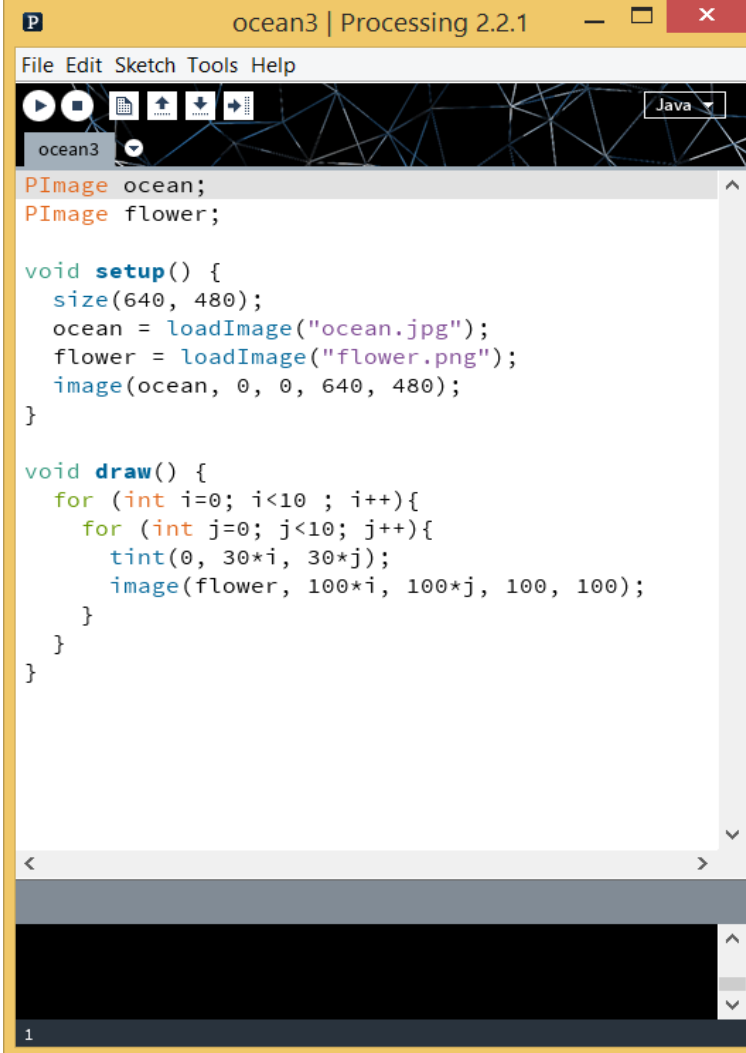
```
// a function that draws n eggs
// one next to the other
void drawEggs(int n) {
    for (int i = 0; i < n; i++) {
        ellipse(100+i*70, 250, 55, 77);
    }
}
```

- Remember the function that drew eggs on the screen? It was just a simple for loop, like the one we did with flowers here.

Magic recipes, ocean3.pde

26

- Download file ocean3.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open ocean3.pde from folder ocean3



```
ocean3 | Processing 2.2.1
File Edit Sketch Tools Help
ocean3
PImage ocean;
PImage flower;

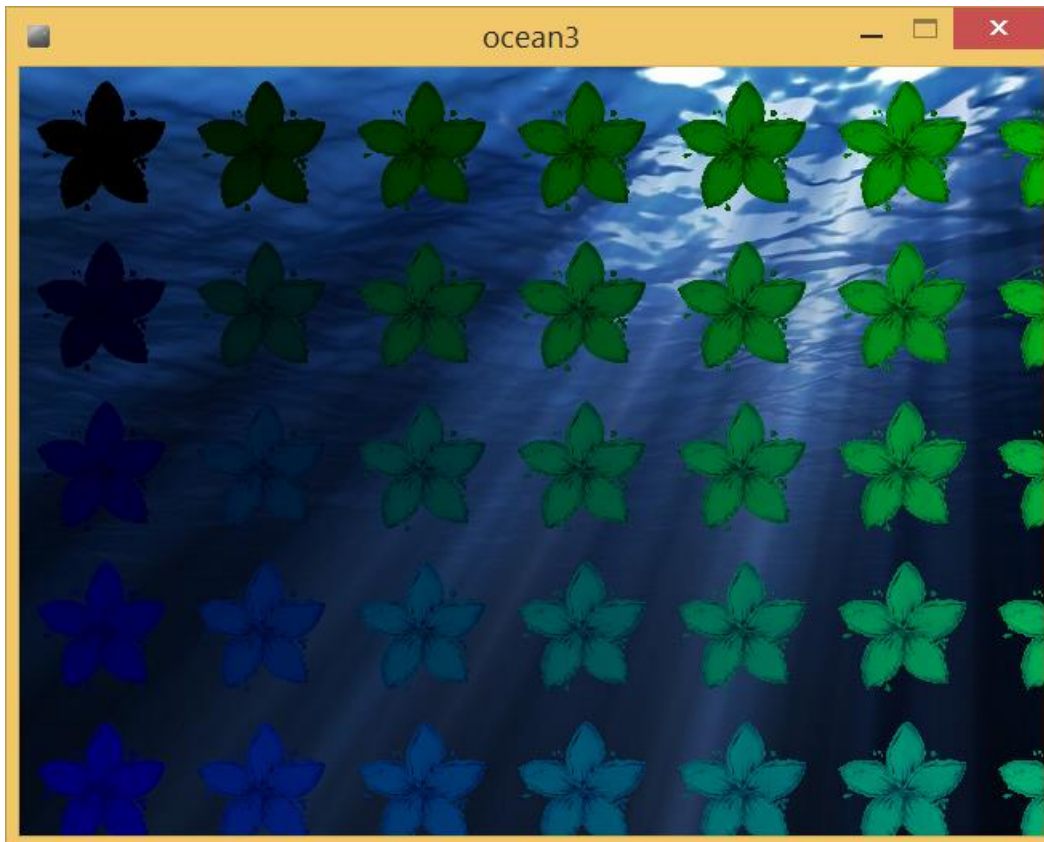
void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

void draw() {
  for (int i=0; i<10 ; i++){
    for (int j=0; j<10; j++){
      tint(0, 30*i, 30*j);
      image(flower, 100*i, 100*j, 100, 100);
    }
  }
}
```

Magic recipes, ocean3.pde

27

- Place even more copies of one image!



```
ocean3 | Processing 2.2.1
File Edit Sketch Tools Help
ocean3
PImage ocean;
PImage flower;

void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
}

void draw() {
  for (int i=0; i<10 ; i++){
    for (int j=0; j<10; j++){
      tint(0, 30*i, 30*j);
      image(flower, 100*i, 100*j, 100, 100);
    }
  }
}
```

Processing – “Nested” For loop

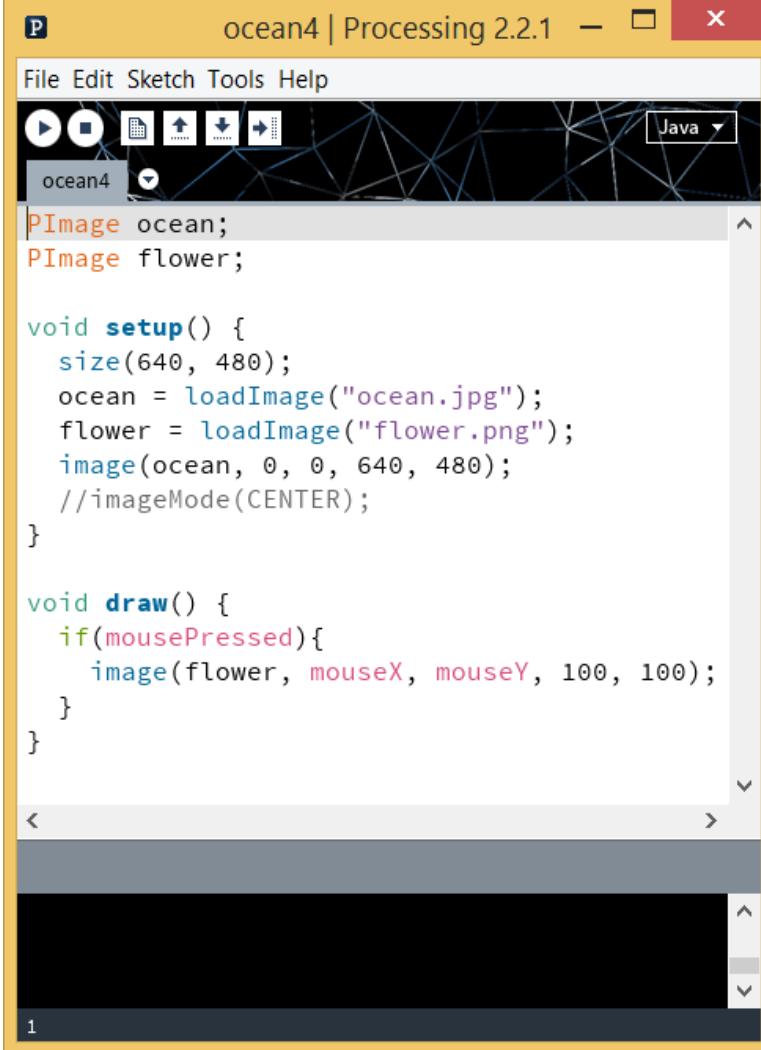
28

- Make processing do many things for you!
- A **for loop** says to Processing “Do this thing N times for me”
 - E.g., Draw 5 images for me, one next to the other
- We can use this as many times we want, even **combine it with another** to do more automatization
 - E.g., Do the previous thing 5 times for me, and each time draw at a lower place

Magic recipes, ocean4.pde

29

- Download file ocean4.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open ocean4.pde from folder ocean4

A screenshot of the Processing IDE window titled "ocean4 | Processing 2.2.1". The window shows a code editor with the following code:

```
PIImage ocean;
PIImage flower;

void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
  //imageMode(CENTER);
}

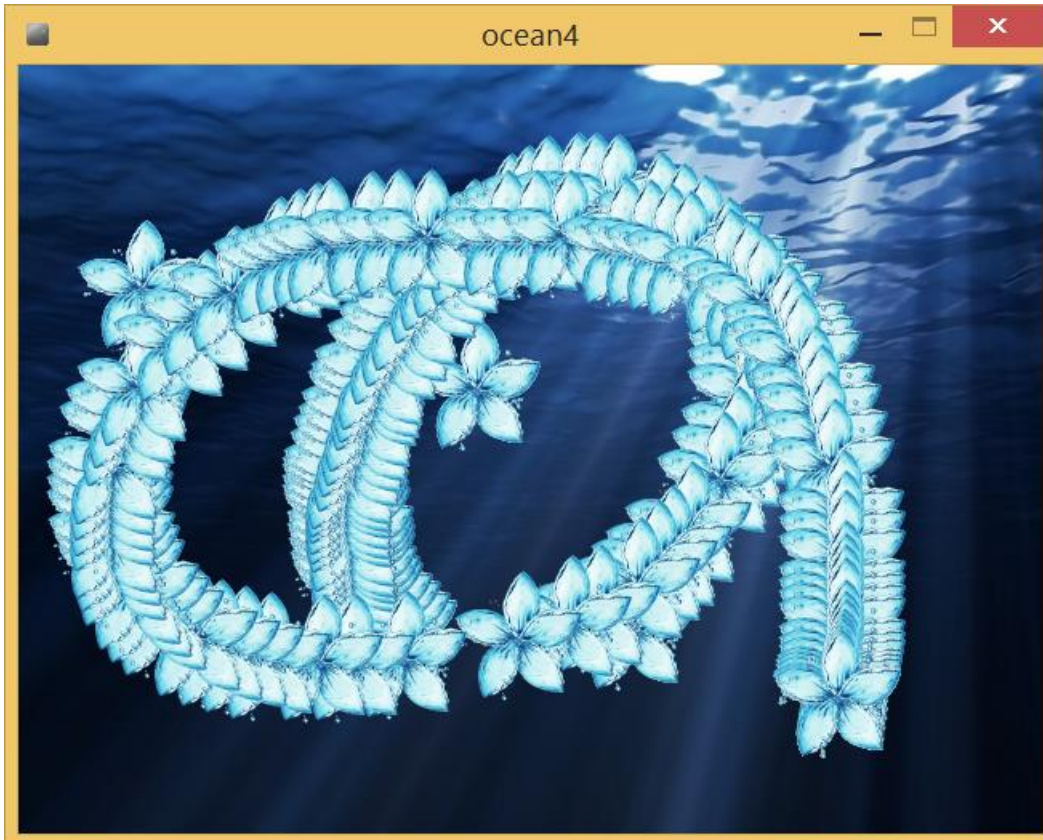
void draw() {
  if(mousePressed){
    image(flower, mouseX, mouseY, 100, 100);
  }
}
```

The IDE interface includes a menu bar (File, Edit, Sketch, Tools, Help), a toolbar with icons for play, stop, and other functions, and a dropdown menu for the language (Java). The code editor has a dark background with syntax highlighting. The status bar at the bottom shows the number "1".

Magic recipes, ocean4.pde

30

- Use the mouse to place copies of an image



```
ocean4 | Processing 2.2.1
File Edit Sketch Tools Help
ocean4
PImage ocean;
PImage flower;

void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
  //imageMode(CENTER);
}

void draw() {
  if(mousePressed){
    image(flower, mouseX, mouseY, 100, 100);
  }
}
1
```

Processing – Mouse input

31

- **mouseX**, **mouseY** are variables that always contain the current x and y position of the mouse in the window
- **mousePressed** says if the main button of the mouse is pressed at the current “frame”

```
void draw() {  
    if (mousePressed) {  
        image (flower, mouseX, mouseY, 100, 100);  
    }  
}
```

Processing – Mouse input

32

- **mouseX**, **mouseY** are variables that always contain the current x and y position of the mouse in the window
- **mousePressed** says if the main button of the mouse is pressed at the current “frame”

```
void draw() {  
    if (mousePressed) {  
        image(flower, mouseX, mouseY, 100, 100);  
    }  
}
```


Processing – Mouse input

33

- **mouseX**, **mouseY** are variables that always contain the current x and y position of the mouse in the window
- **mousePressed** says if the main button of the mouse is pressed at the current “frame”

```
void draw() {  
  if (mousePressed) {  
    image(flower, mouseX, mouseY, 100, 100);  
  }  
}
```

Processing – Mouse input

34

- **mouseX**, **mouseY** are variables that always contain the current x and y position of the mouse in the window
- **mousePressed** says if the main button of the mouse is pressed at the current “frame”

```
void draw() {  
    if (mousePressed) {  
        image (flower, mouseX, mouseY, 100, 100);  
    }  
}
```

Processing – Images

35

- Notice that the image of the flower is not placed exactly at the place where you click – Why?

Processing – Images

36

- Notice that the image of the flower is not placed exactly at the place where you click – Why?
- There are **two ways to place an image**, the default one is to place it so that the image upper-left corner is located at the specified x,y position

```
imageMode (CORNER) ;
```

- But you can also change this so that the image center is located at the specified x,y position

```
imageMode (CENTER) ;
```

Processing – Images

37

- All of these statements are like “putting a brush into the bucket with the paint and then paint”, i.e., they affect all later statements

```
imageMode (CORNER) ;
```

```
imageMode (CENTER) ;
```

```
tint (0, 255, 0, 30) ;
```

```
tint (255, 0, 0, 30) ;
```



```
image (ocean, 0, 0, 640, 480) ;
```

```
image (flower, 0, 0, 100, 100) ;
```

Processing – Images

38

- Note that we normally want to deal with background images differently than images we want to place on a particular point, e.g.,

```
imageMode (CENTER) ;
```

```
tint (255, 0, 0, 30) ;
```



```
image (flower, 0, 0, 100, 100) ;
```

Processing – Images

39

- Note that we normally want to deal with background images differently than images we want to place on a particular point, e.g.,

```
imageMode (CORNER) ;
```

```
tint (255, 255, 255, 255) ;
```



```
image (ocean, 0, 0, 640, 480) ;
```

Processing – Images

40

- This is the same as with drawing circles and rectangles, there we typically want to specify the **fill color** and the **stroke color** of drawings

```
fill (255, 255, 255) ;
```

```
fill (0, 0, 0) ;
```

```
stroke (0, 255, 0) ;
```

```
stroke (255, 0, 0) ;
```



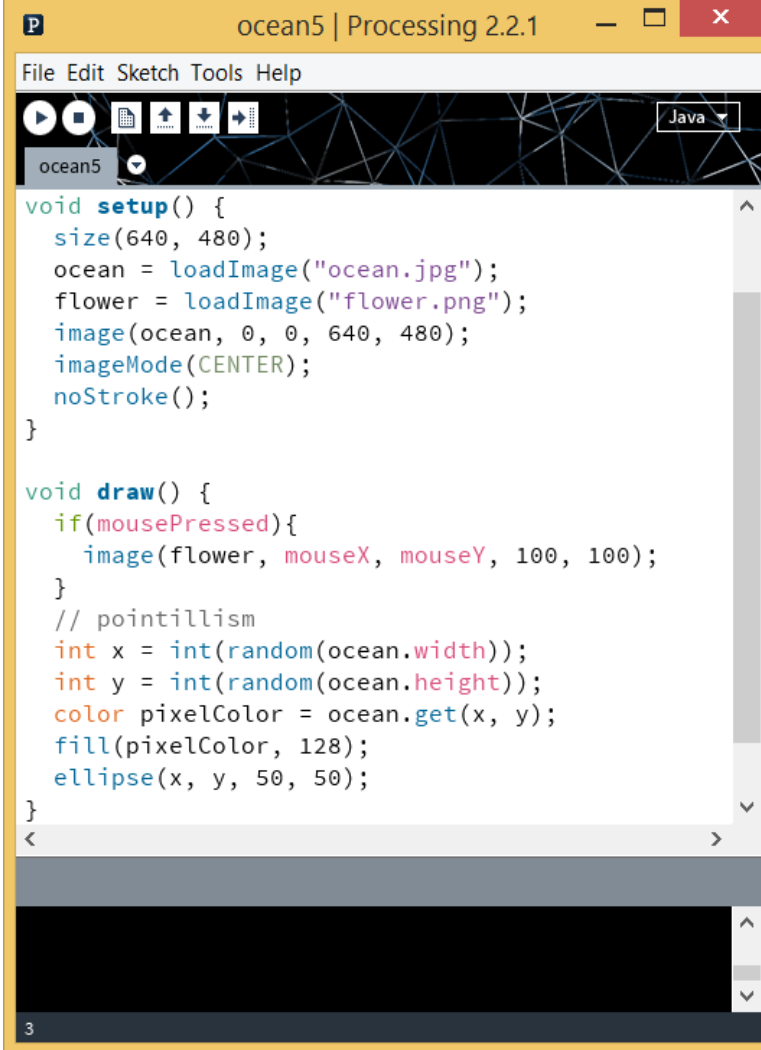
```
rect (x1, y1, x2, y2) ;
```

```
ellipse (x, y, width, height) ;
```


Magic recipes, ocean5.pde

41

- Download file ocean5.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open ocean5.pde from folder ocean5



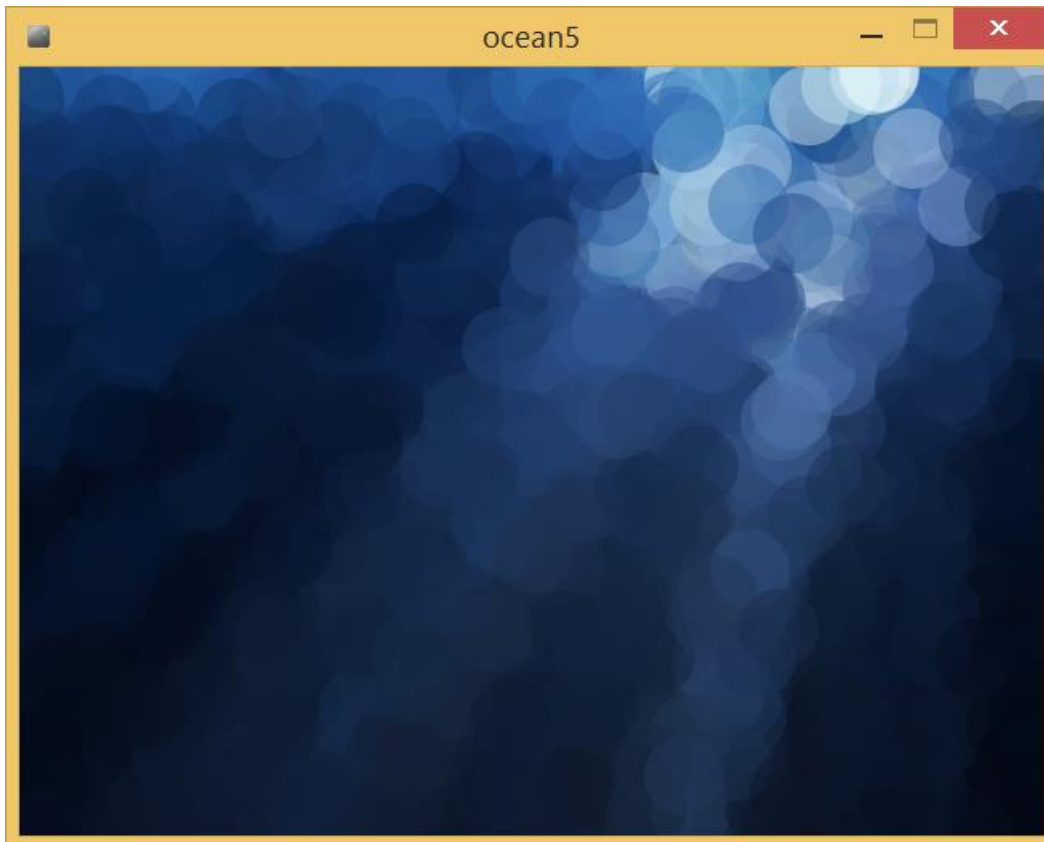
```
ocean5 | Processing 2.2.1
File Edit Sketch Tools Help
ocean5
void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
  imageMode(CENTER);
  noStroke();
}

void draw() {
  if(mousePressed){
    image(flower, mouseX, mouseY, 100, 100);
  }
  // pointillism
  int x = int(random(ocean.width));
  int y = int(random(ocean.height));
  color pixelColor = ocean.get(x, y);
  fill(pixelColor, 128);
  ellipse(x, y, 50, 50);
}
3
```

Magic recipes, ocean5.pde

42

- Place circles at random points following image color



```
ocean5 | Processing 2.2.1
File Edit Sketch Tools Help
ocean5
void setup() {
  size(640, 480);
  ocean = loadImage("ocean.jpg");
  flower = loadImage("flower.png");
  image(ocean, 0, 0, 640, 480);
  imageMode(CENTER);
  noStroke();
}

void draw() {
  if(mousePressed){
    image(flower, mouseX, mouseY, 100, 100);
  }
  // pointillism
  int x = int(random(ocean.width));
  int y = int(random(ocean.height));
  color pixelColor = ocean.get(x, y);
  fill(pixelColor, 128);
  ellipse(x, y, 50, 50);
}
3
```

Processing – Mouse input

43

```
float x = random(10);
```

- The variable **x** is assigned a new value that is randomly chosen from 0 to 10
- Here, **x,y** is a random point in the window

```
void draw() {  
    float x = random(640);  
    float y = random(480);  
    ellipse(x, y, 50, 50);  
}
```

Processing – Mouse input

44

- We can use `random()` with more specific values, e.g., the size of an image in the window, and then even pick the color of the point and draw using that color

```
void draw() {  
    float x = random(ocean.width);  
    float y = random(ocean.height);  
    color pixel = ocean.get(x, y);  
    tint(pixel);  
    ellipse(x, y, 50, 50);  
}
```

Magic recipes, ocean6.pde

45

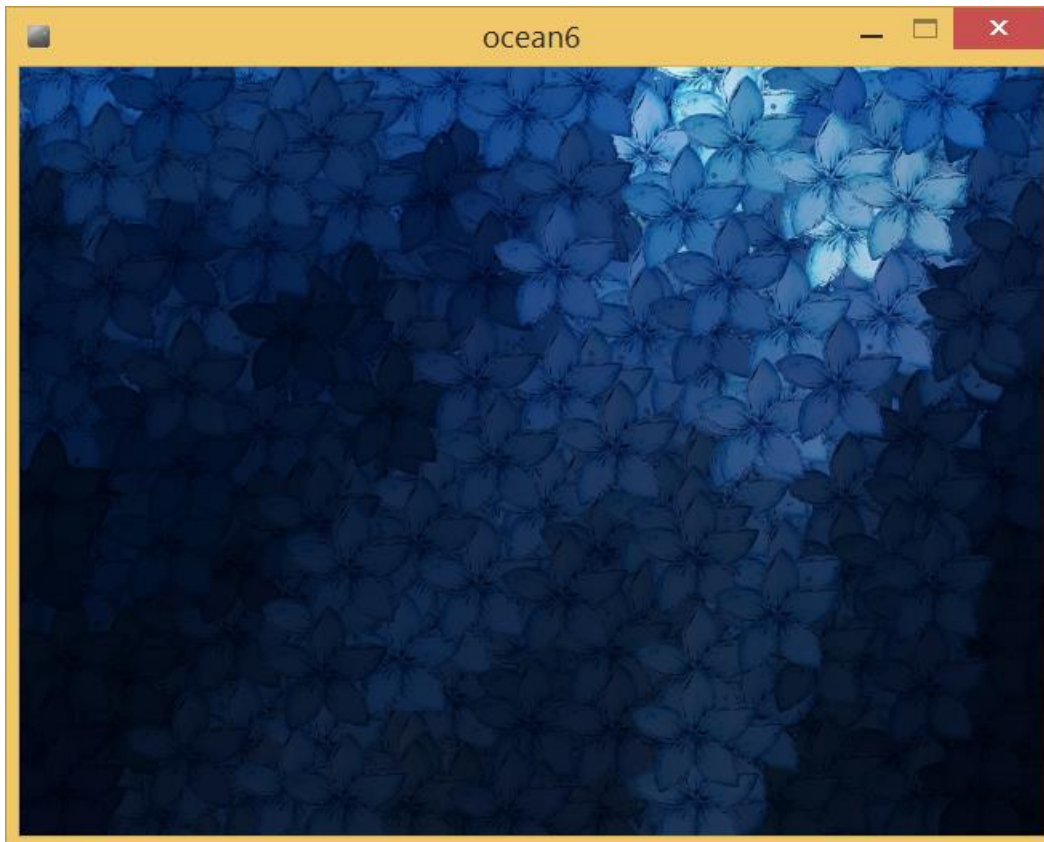
- Download file ocean6.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open ocean6.pde from folder ocean6

```
void draw() {
  if(mousePressed){
    color pixelColor = ocean.get(mouseX, mouseY);
    tint(pixelColor);
    image(flower, mouseX, mouseY, 100, 100);
  }
  // pointillism
  int x = int(random(ocean.width));
  int y = int(random(ocean.height));
  color pixelColor = ocean.get(x, y);
  tint(pixelColor);
  image(flower, x, y, 100, 100);
}
```

Magic recipes, ocean6.pde

46

- Place image at random points following image color

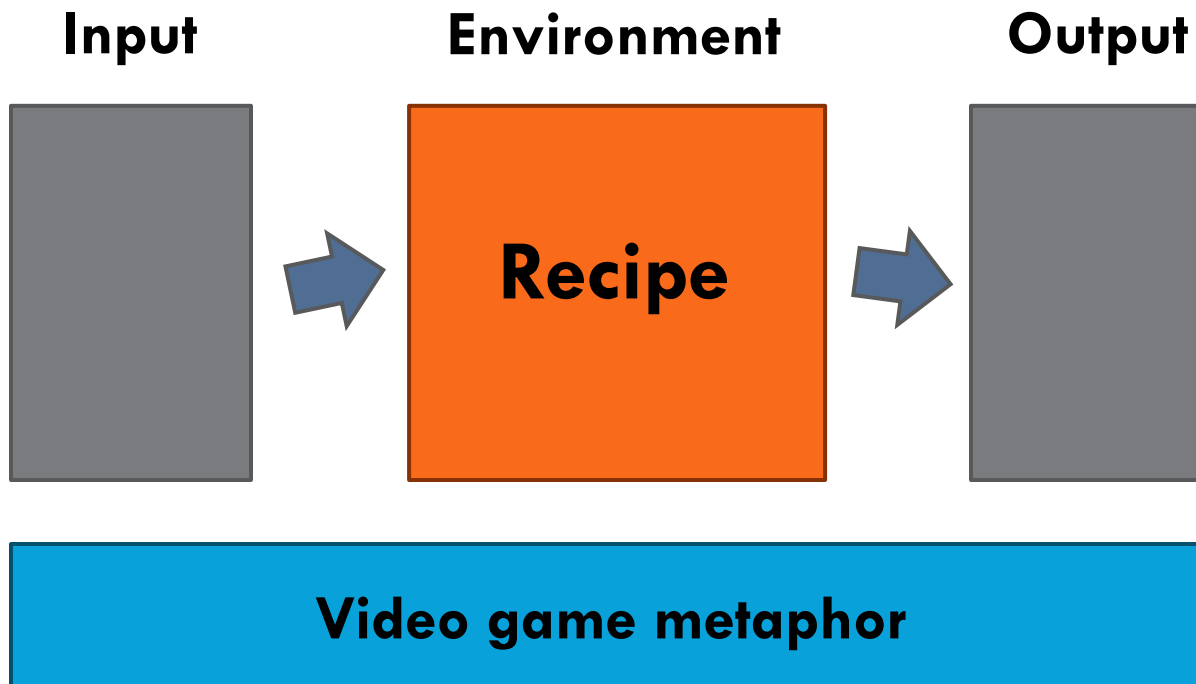


```
ocean6 | Processing 2.2.1
File Edit Sketch Tools Help
ocean6
void draw() {
  if(mousePressed){
    color pixelColor = ocean.get(mouseX, mouseY);
    tint(pixelColor);
    image(flower, mouseX, mouseY, 100, 100);
  }
  // pointillism
  int x = int(random(ocean.width));
  int y = int(random(ocean.height));
  color pixelColor = ocean.get(x, y);
  tint(pixelColor);
  image(flower, x, y, 100, 100);
}
29
```

Interactive Multimedia Design

47

- It's important to separate the inputs and outputs



Interactive Multimedia Design

48

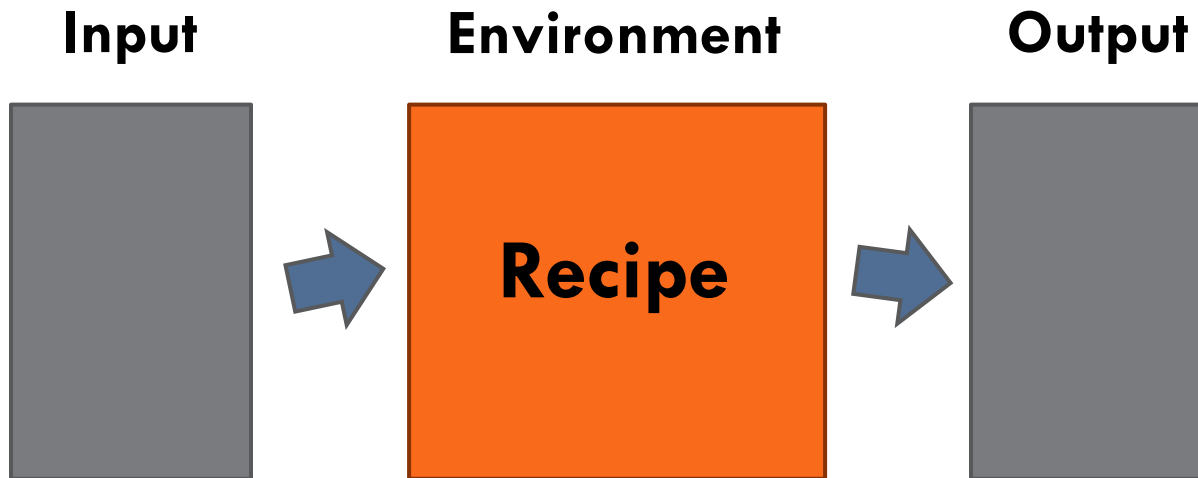
- It's important to separate the inputs and outputs
- We “monitor” the input, i.e., the mouse click, and then we generate output, i.e., flowers on a background, based on the mouse position, which is also an input



Interactive Multimedia Design

49

- It's important to separate the inputs and outputs
 - ▣ Input: Mouse position, Mouse click, Yo message (IOTUP, IOTDOWN)
 - ▣ Output: Drawings, Sound



Interactive Multimedia Design

50

- It's important to separate the inputs and outputs
 - ▣ Input: Mouse position, Mouse click, Yo message (IOTUP, IOTDOWN), **chat!**
 - ▣ Output: Drawings, Sound, **chat!**



Reminder: Processing reference

51

- ▣ https://processing.org/reference/color_datatype.html
- ▣ <https://processing.org/reference/PImage.html>
- ▣ <https://processing.org/reference/loadImage.html>
- ▣ “Processing Reference” is like a **Spell Book!**
 - ▣ <http://processing.org/reference/>
- ▣ There you can find all available magic words you can use as well as a detailed explanation of the intended use and examples

Interactive Multimedia Design

52

```
void draw() {  
    if (mousePressed) {  
        fill(alert);  
    } else {  
        fill(gray);  
    }  
    rect(50, 50, 250, 250, 17);  
    image(img, 100, 100, 150, 150);  
}
```

- New magic word: **else** !